CLAIMS

1. A method for storage of security keys and certificates in a data processing system comprising:

providing at least one entity (150) in the form of a key or certificate for storage in a storage means;

fragmenting the entity into fragments (152, 154) of non-uniform length according to a predetermined algorithm (200);

storing the fragments (152, 154) in the storage means (280);

wherein fragments (152, 154) of the at least one entity (150) are intermixed within the storage means.

2. A method for storage as claimed in claim 1, wherein the storage means is a data file including a block of data (110) accommodating the entities (150).

3. A method for storage as claimed in claim 1, wherein the storage means also contains random bit patterns (120).

4. A method for storage as claimed in claim 1, wherein the step of fragmenting the entity (150), fragments the bytes of the entity (150).

5. A method for storage as claimed in claim 1, wherein the location of storing the fragments (152, 154) is also determined by the algorithm (200).

6. A method for storage as claimed in claim 1, wherein the entity (150) can be read from the storage means by using the algorithm (200) to find and recombine the fragments (152, 154) of the entity (150).

7. A method for storage as claimed in claim 1, wherein the storage means has a pass code (140) and the algorithm (200) for fragmenting uses the pass code (140).

8. A method for storage as claimed in claim 7, wherein the fragments (152, 154) are stored at locations in the storage means determined by using the pass code (140).

9. A method for storage as claimed in claim 1, wherein the method includes keeping a bit map (130) as a record of fragment locations until the storage is complete (190).

10. A method for storage as claimed in claim 1, wherein in the event that a fragment (152) has already been stored at a location required for a subsequent fragment (154), the subsequent fragment (154) is stored immediately after the existing fragment (152).

11. A method for storage as claimed in claim 1, wherein the storage means is a Java keystore repository.

12. A method for storage as claimed in claim 11, wherein the algorithm (200) is implemented as a Java keystore class.

13. An apparatus for storage of security keys and certificates in a data processing system comprising:
a storage means;
at least one entity (150) in the form of a key or certificate for storage in the storage means;
wherein the entity (150) is stored in fragments (152, 154) of non-uniform length according to a predetermined algorithm (200) and fragments of the at least one entity (150) are intermixed within the storage means.

14. An apparatus for storage as claimed in claim 13, wherein the storage means is a data file including a block of data (110) accommodating the entities (150).

15. An apparatus for storage as claimed in claim 13, wherein the storage means also contains random bit patterns (120).

16. An apparatus for storage as claimed in claim 13, wherein the location of the fragments (152, 154) is also determined by the algorithm (200).

17. An apparatus for storage as claimed in claim 13, wherein the entity (150) can be read from the storage means by using the algorithm (200) to find and recombine the fragments (152, 154) of the entity (150).

18. An apparatus for storage as claimed in claim 13, wherein the storage means has a pass code (140) and the algorithm (200) for fragmenting uses the pass code (140).

19. An apparatus for storage as claimed in claim 18, wherein the fragments (152, 154) are stored at locations in the storage means determined by using the pass code (140).

20. An apparatus for storage as claimed in claim 13, wherein a bit map (130) is kept as a record of fragment locations until the storage is complete (190).

21. An apparatus for storage as claimed in claim 13, wherein the storage means is a Java keystore repository.

22. An appartus for storage as claimed in claim 21, wherein the algorithm (200) is implemented as a Java keystore class.

23. A computer program product for storage of security keys and certificates in a data processing system, said product comprising program instructions in machine-readable form on a medium, said instructions causing the system to perform the steps of:

providing at least one entity (150) in the form of a key or certificate for storage in a storage means;

fragmenting the entity into fragments (152, 154) of non-uniform length according to a predetermined algorithm (200);

storing the fragments (152, 154) in the storage means (280);
wherein fragments (152, 154) of the at least one entity (150) are intermixed within the storage means.

24. A computer program product for storage as claimed in claim 23, wherein the storage means is a data file including a block of data (110) accommodating the entities (150).

25. A computer program product for storage as claimed in claim 23, wherein the storage means also contains random bit patterns (120).

26. A computer program product for storage as claimed in claim 23, wherein the step of fragmenting the entity (150), fragments the bytes of the entity (150).

27. A computer program product for storage as claimed in claim 23, wherein the location of storing the fragments (152, 154) is also determined by the algorithm (200).

28. A computer program product for storage as claimed in claim 23, wherein the entity (150) can be read from the storage means by using the algorithm (200) to find and recombine the fragments (152, 154) of the entity (150).

29. A computer program product for storage as claimed in claim 23, wherein the storage means has a pass code (140)

and the algorithm (200) for fragmenting uses the pass code (140).

30. A computer program product for storage as claimed in claim 29, wherein the fragments (152, 154) are stored at locations in the storage means determined by using the pass code (140).

31. A computer program product for storage as claimed in claim 23, wherein the instructions further cause the system to keep a bit map (130) as a record of fragment locations until the storage is complete (190).

32. A computer program product for storage as claimed in claim 23, wherein in the event that a fragment (152) has already been stored at a location required for a subsequent fragment (154), the subsequent fragment (154) is stored immediately after the existing fragment (152).

33. A computer program product for storage as claimed in claim 23, wherein the storage means is a Java keystore repository.

34. A computer program product for storage as claimed in claim 33, wherein the algorithm (200) is implemented as a Java keystore class.